

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2001-230770

(P2001-230770A)

(43) 公開日 平成13年8月24日 (2001.8.24)

(51) Int.Cl. ⁷	識別記号	F I	ターマート* (参考)
H 0 4 L 9/10		G 0 9 C 1/00	6 2 0 Z 5 J 1 0 4
G 0 9 C 1/00	6 2 0	H 0 4 L 9/00	6 2 1 A 9 A 0 0 1

審査請求 未請求 請求項の数20 O L (全 22 頁)

(21) 出願番号 特願2000-35898(P2000-35898)

(22) 出願日 平成12年2月14日 (2000.2.14)

(71) 出願人 000003078

株式会社東芝

神奈川県川崎市幸区堀川町72番地

(72) 発明者 橋本 幹生

神奈川県川崎市幸区小向東芝町1 株式会
社東芝研究開発センター内

(72) 発明者 斉藤 健

神奈川県川崎市幸区小向東芝町1 株式会
社東芝研究開発センター内

(74) 代理人 100083806

弁理士 三好 秀和 (外7名)

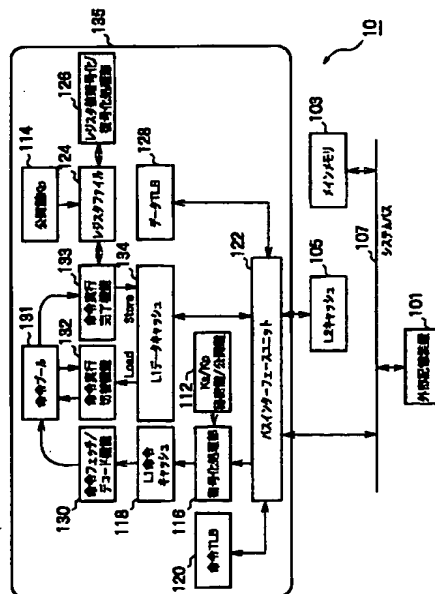
最終頁に続く

(54) 【発明の名称】 マイクロプロセッサ

(57) 【要約】

【課題】 実行プログラムの不正な書き換えを確実に防止することのできるマイクロプロセッサを提供する。

【解決手段】 マイクロプロセッサは、外部へ読み出し不可能な固有の秘密鍵を内部に保持し、この秘密鍵に対応する固有の公開鍵であらかじめ暗号化された内容を復号化する。マイクロプロセッサのバスインターフェイスユニットは、仮想アドレスを物理アドレスに変換するアドレス変換手段と、マイクロプロセッサ外部のメモリから指定された物理アドレスに対応する内容を読み出すメモリ読み出し手段とを含む。命令TLBは、アドレス変換規則と、各々が仮想アドレスで指定された範囲の暗号化属性情報を有する1以上のエントリとを含むテーブルを格納する。復号化処理部は、暗号化属性情報が、仮想アドレスで指定された範囲が暗号化されていることを示す場合に、対応する内容をメモリ読み出し手段を介して外部メモリから読み出し、復号化する。



【特許請求の範囲】

【請求項1】 外部へ読み出すことのできない固有の秘密鍵を内部に保持し、この秘密鍵に対応する公開鍵によってあらかじめ暗号化された内容を復号化する1チップまたは1パッケージのマイクロプロセッサであって、仮想アドレスを物理アドレスに変換するアドレス変換手段と、

前記アドレス変換手段による仮想アドレスから物理アドレスへの変換規則と、各々が前記仮想アドレスで指定された範囲の暗号化属性情報を有する1以上のエン트리とを含むテーブルを格納する第1の記憶手段と、

マイクロプロセッサ外部のメモリから、変換された物理アドレスに対応する内容を読み出すメモリ読み出し手段と、

前記テーブルのエントリに含まれる前記暗号化属性情報が、当該仮想アドレスで指定された範囲が暗号化されていることを示す場合に、前記外部のメモリに記録された前記内容を、前記メモリ読み出し手段を介して読み出し、読み出した内容を前記秘密鍵によって復号化する復号化手段と、

前記復号化された内容を一時的に記憶する第2の記憶手段と、

前記第2の記憶手段に記憶された内容を逐次解釈する命令デコード手段とを具備したことを特徴とするマイクロプロセッサ。

【請求項2】 前記第2の記憶手段に格納された内容は、前記命令デコード手段以外からは読み出し禁止とすることを特徴とする請求項1に記載のマイクロプロセッサ。

【請求項3】 前記仮想アドレスで指定されている前記外部のメモリの対応する物理アドレスに対して読み出しがあった場合に、この読み出しに対して例外処理を発生させるか、前記対応する物理アドレスに記録されている値とは異なる値を回答するかの、いずれかの処理を行うことを特徴とする請求項1に記載のマイクロプロセッサ。

【請求項4】 前記各エントリは、前記仮想アドレスで指定された範囲のデバッグ属性情報を含み、当該デバッグ属性情報が、指定された範囲でデバッグ作業が行われていることを示しているときに、前記外部のメモリの対応する物理アドレスに対して読み出しまたは書き込みがあった場合に、前記読み出しまたは書き込みに対して、例外処理を発生させることを特徴とする請求項1に記載のマイクロプロセッサ。

【請求項5】 前記暗号化属性情報が、前記仮想アドレスで指定された範囲が暗号化されていないことを示した場合に、前記第2の記憶手段に格納した復号化された内容を破棄することを特徴とする請求項1に記載のマイクロプロセッサ。

【請求項6】 前記復号化手段によって復号化された前

記プログラムの実行状態の値を記憶するレジスタと、前記暗号化プログラムの実行中に割り込みが発生した場合に、前記レジスタに記憶されている値を、ランダムな値を持つ任意の暗号化鍵で暗号化するレジスタ暗号化／復号化処理部とをさらに具備することを特徴とする請求項1～5のいずれかに記載のマイクロプロセッサ。

【請求項7】 前記レジスタ値復号化／暗号化処理部は、前記レジスタ値の暗号化に用いた任意の暗号化鍵を、前記固有の秘密鍵でさらに暗号化し、暗号化された任意の暗号化鍵を前記マイクロプロセッサ外部のメモリに格納させることを特徴とする請求項6に記載のマイクロプロセッサ。

【請求項8】 前記レジスタ値暗号化／復号化処理部は、前記割り込みにより中断された暗号化プログラムの実行が再開される場合に、前記暗号化された実行状態の値を復号化することを特徴とする請求項6に記載のマイクロプロセッサ。

【請求項9】 前記レジスタ値暗号化／復号化処理部は、前記割り込みが実行中の暗号化プログラム自体によって発生されたシステムコールである場合には、前記レジスタに記憶されている実行状態の値の暗号化は行わないことを特徴とする請求項6に記載のマイクロプロセッサ。

【請求項10】 外部へ読み出すことのできない固有の秘密鍵と、この秘密鍵に対応する公開鍵とを内部に保持し、任意の共通鍵によって暗号化された内容を復号化する1チップまたは1パッケージのマイクロプロセッサであって、

仮想アドレスを物理アドレスに変換するアドレス変換手段と、

前記アドレス変換手段による仮想アドレスから物理アドレスへの変換規則と、前記公開鍵によってあらかじめ外部で暗号化された前記共通鍵についての情報とを含むテーブルを格納する第1の記憶手段と、

マイクロプロセッサ外部のメモリから、指定された物理アドレスに対応する内容を読み出すメモリ読み出し手段と、

前記暗号化された共通鍵を、前記秘密鍵で復号する第1の復号化処理手段と、

前記暗号化された内容を、前記復号化された共通鍵で復号化する第2の復号化処理手段と、

前記復号化された内容を一時的に記憶する第2の記憶手段と、

前記第2の記憶手段に記憶された内容を逐次解釈する命令デコード手段とを具備することを特徴とするマイクロプロセッサ。

【請求項11】 外部へ読み出すことのできない固有の秘密鍵とこの秘密鍵に対応する固有の公開鍵とをあらかじめ有するマイクロプロセッサが内蔵されたクライアント装置に、ネットワークを介して実行プログラムを配布

するプログラム配布装置であって、
前記クライアント装置との間に第1の通信路を設定する第1通信路設定部と、
前記第1の通信路を介して前記クライアント装置を使用するユーザの認証を行うユーザ認証部と、
前記第1の通信路上に、クライアント装置が内蔵するマイクロプロセッサに直接連絡する第2の通信路をさらに設定する第2通信路設定部と、
前記クライアント装置に配布すべき実行プログラムを暗号化して暗号化プログラムを作成する暗号化処理部と、
前記暗号化したプログラムを前記第2の通信路を介してクライアント装置のマイクロプロセッサに送信する配布送信部とを備えたプログラム配布装置。

【請求項12】 前記第2の通信路を介して、クライアント装置のマイクロプロセッサが正当であることを認証するプロセッサ認証部をさらに有し、前記プロセッサ認証部は、前記第2の通信路を介して、クライアント装置のマイクロプロセッサが、該マイクロプロセッサに固有の公開鍵と秘密鍵とを確かに保持することを認証することを特徴とする請求項11に記載のプログラム配布装置。

【請求項13】 前記暗号化処理部は、前記第2の通信路を介して前記クライアント装置のマイクロプロセッサから固有の公開鍵を受け取り、該公開鍵を用いて前記実行プログラムを暗号化することを特徴とする請求項11または12に記載のプログラム配布装置。

【請求項14】 前記暗号化処理部は、任意の暗号化鍵で前記実行プログラムを暗号化し、前記第2の通信路を介してクライアント装置のマイクロプロセッサから固有の公開鍵を受け取り、前記任意の暗号化鍵を前記公開鍵で暗号化し、
前記配布送信部は、前記暗号化した実行プログラムと、暗号化された暗号化鍵とを、前記第2の通信路を介してクライアント装置のマイクロプロセッサに送信することを特徴とする請求項11または12に記載のプログラム配布装置。

【請求項15】 ネットワークを介してプログラム配布装置からプログラムの配布を受けるクライアント装置であって、
あらかじめ固有の秘密鍵と公開鍵とを有するマイクロプロセッサと、
前記プログラム配布装置との間に第1の通信路を設定する第1クライアント側通信路設定部と、
前記第1の通信路を介して、前記クライアント装置を使用するユーザのユーザIDを前記プログラム配布装置に送信するユーザ認証部と、
前記第1の通信路上に、前記マイクロプロセッサから直接プログラム配布装置に連絡する第2の通信路をさらに設定する第2クライアント側通信路設定部と、
前記第2の通信路を介してプログラム配布装置に、前記

マイクロプロセッサが該マイクロプロセッサに固有の秘密鍵と公開鍵とを確かに保持することを証明する証明を送信する証明部と、

前記第2の通信路を介して前記プログラム配布装置から、暗号化された実行プログラムを受信する受信部とを有することを特徴とするクライアント装置。

【請求項16】 ネットワークと、
前記ネットワークに接続され、実行プログラムを前記ネットワークを介して配布するプログラム配布装置と、
前記ネットワークに接続され、ネットワークを介して前記プログラム配布装置から実行プログラムの配布を受けるクライアント装置とを含み、
前記クライアント装置は、
あらかじめ固有の秘密鍵と公開鍵とを有するマイクロプロセッサと、
前記プログラム配布装置との間に第1の通信路を設定する第1クライアント側通信路設定部と、
前記第1の通信路上に、前記マイクロプロセッサから直接プログラム配布装置に連絡する第2の通信路をさらに設定する第2クライアント側通信路設定部と、前記第2の通信路を介してプログラム配布装置に、公開鍵および該公開鍵が確か前記マイクロプロセッサに固有のものであることを証明する証明を送信する証明部と、前記第2の通信路を介して前記プログラム配布装置から、実行プログラムを受信する受信部とを備え、
前記プログラム配布装置は、
前記クライアント装置との間に第1の通信路を設定する第1通信路設定部と、
前記第1の通信路を介して前記クライアント装置を使用するユーザの認証を行うユーザ認証部と、
前記第1の通信路上に、クライアント装置が内蔵するマイクロプロセッサに直接連絡する第2の通信路をさらに設定する第2通信路設定部と、
前記クライアント装置に配布すべき実行プログラムを暗号化して暗号化プログラムを作成する暗号化処理部と、
前記暗号化したプログラムを前記第2の通信路を介してクライアント装置のマイクロプロセッサに直接送信する配布送信部とを備えることを特徴とするプログラム配布システム。

【請求項17】 固有の秘密鍵と、この秘密鍵に対応する固有の公開鍵とをあらかじめ有するマイクロプロセッサを内蔵するコンピュータから、プログラム配布装置に、ネットワークを介して第1の通信路を設定するステップと、
前記第1の通信路上に、前記マイクロプロセッサからプログラム配布装置に直接連絡する第2の通信路をさらに設定するステップと、
前記第2の通信路を介して、前記マイクロプロセッサから前記プログラム配布装置に、前記固有の公開鍵を送信するステップと、
前記ネットワークを介して送信された公開鍵を、プログ

10

20

30

40

50

ラム配布装置で受信し、プログラム配布装置で配布すべきプログラムを暗号化するステップと、
前記暗号化したプログラムを、前記第2の通信路を介して、前記コンピュータのマイクロプロセッサに直接送信するステップと、
を含むプログラム配布方法。

【請求項18】 あらかじめ固有の秘密鍵と、この秘密鍵に対応する固有の公開鍵とを有するマイクロプロセッサで実行される暗号化プログラムのロード方法であって、

前記暗号化プログラムに対して1以上のページからなる仮想記憶領域を割り当てるステップと、

前記1以上のページに前記暗号化プログラムを暗号化されたままの状態書き込むステップと、

前記書き込んだページに対応するページテーブルエントリの暗号化フラグをセットするステップとを含む暗号化プログラムのロード方法。

【請求項19】 前記1以上のページのいずれかのページに書き込まれた暗号化プログラムが前記マイクロプロセッサに読み込まれ、前記秘密鍵で復号化されて実行された場合に、前記読み込まれたページに対応するページテーブルエントリの暗号化フラグをリセットするステップと、

前記暗号化フラグのリセットに応じて、前記読み込まれたページの内容をクリアして、このページを開放するステップとをさらに含むことを特徴とする請求項18に記載のプログラムロード方法。

【請求項20】 コンピュータシステムのメモリに読み込まれて、前記コンピュータシステムが内蔵するマイクロプロセッサによって実行されるファイルを記録したコンピュータ読み取り可能な記録媒体であって、

あらかじめ所定の暗号化鍵で暗号化された実行ファイルを記録し、外部のプログラムへの直接の参照を含まない1以上の第1プログラム領域と、

外部のプログラムへの直接の参照を含む1以上の第2プログラム領域とを有し、前記第2プログラム領域は、外部のプログラムモジュールへのジャンプ命令を並べたテーブルを平文の状態記録し、これにより参照先のアドレスに応じてジャンプ先の書き換えを可能にすることを特徴とする実行ファイルを記録したコンピュータ読み取り可能な記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は、パーソナルコンピュータなどのハードウェアに組み込まれて暗号化されたプログラムを実行するマイクロプロセッサ、このマイクロプロセッサを組み込んだハードウェア装置にインターネット等の公衆ネットワークを通じて暗号化された実行プログラムを配布するプログラム配布システムと配布方法、および配布された実行プログラムをハードウェア内

の既存のオペレーティングシステムと整合してロードすることのできる暗号化ファイルロード方法に関する。

【0002】

【従来の技術】 近年、マイクロプロセッサの性能向上は著しく、従来の機能である計算やグラフィックだけではなく、動画像や音声の再生、編集加工が可能になっている。このようなマイクロプロセッサをエンドユーザ向けシステム（以下、「PC」と呼ぶ。）に組み込むことによって、ユーザはモニタ上でさまざまな動画像や音声を楽しめるようになった。また、動画像や音声の再生機能を、PC本来の計算能力と組み合わせることによって、ゲームなどへの応用を高めることができる。このようなマイクロプロセッサは固有のハードウェアを必要とするわけではなく、さまざまなハードウェアに組み込まれ得るので、すでにPCを持っているユーザは、プログラムを実行するマイクロプロセッサを取り代えるだけで、安価に動画像や音声の再生、編集が楽しめるという利点もある。

【0003】 PCで画像や音声を扱う場合に問題となるのが、オリジナルの画像や音楽の著作権の保護である。MDやデジタルビデオデッキでは、これらの装置に不正なコピーを防ぐための機構をあらかじめ組み込むことによって、無制限なコピーを防止することができる。これらの装置を分解、改造してまでも不正コピーを行うことは極めて稀であり、仮にそのような装置があったとしても、世界的にみて、不正コピーを目的として改造された装置の製造、販売を法律によって禁じられる方向にある。したがって、ハードウェアの面での不正コピーによる被害は、それほど問題とはなっていない。

【0004】 しかし、PC上で画像データや音楽データを扱うのは、ハードウェアではなく、ソフトウェアそのものである。そして、エンドユーザはPC上でソフトウェアの改変を自由に行うことができる。すなわち、ユーザにある程度の知識があれば、プログラムを解析することにより、実行ソフトウェアを書き換えて、不正コピーを行い得る可能性は充分にある。さらに、このようにして作られた不正コピー用のソフトウェアは、ハードウェアと異なり、ネットワークなどの媒体を通じて、またたく間に広がってしまうという問題がある。

【0005】 このような問題を解決するために、著作権が問題となる商業映画や音楽の再生に使われるPCソフトウェアには、ソフトウェア自体を暗号化するなどの手法により、解読、改竄されることを防止する技術が用いられている。この技術は、耐タンパソフトウェア技術と呼ばれている（David Aucsmith, et al.: "Tamper Resistant Software: An Implementation", Proceeding of the 1996 Intel Software Developer's Conference）。耐タンパソフトウェア技術は、動画像や音声を始めとしてPCを通じてユーザに提供される著作物や、ノウハウとして価値のある情報が不正にコピーされるのを防止す

るだけではなく、PCソフトウェア自体に含まれるノウハウなどを解析から守るのにも有効である。

【0006】

【発明が解決しようとする課題】しかしながら、耐タンバソフトウェア技術の根本原理は、プログラムの中で保護を要する部分を実行前は暗号化しておき、実行する直前に復号し、実行終了時に再び暗号化することにより、逆アセンブラ、デバッガなどの解析ツールによる解析を困難にするものである。したがって、プログラムがプロセッサによって実行可能である以上、プログラム開始時から順を追って解析していけば必ず解析することが可能である。換言すれば、耐タンバソフトウェア技術を用いても、プログラムコードを逐次的に解析することによって、プログラムの動作が解読可能となる。

【0007】この事実、PCを利用して動画や音声再生するシステムに、著作権者が著作物を供給する際の妨げとなる。また、PCを通じて高度な情報サービスや企業、個人のノウハウを含んだプログラムをPCに適用しようにも、不正コピーの可能性が依然存在する以上、適用が見合わせられ、PCソフトウェアの応用範囲が著しく狭められるという問題もある。

【0008】そこで本発明の第1の目的は、実行プログラムの書き換えを確実に防止することができ、PCソフトウェアの広範囲な適用を可能にするマイクロプロセッサの提供にある。

【0009】本発明の第2の目的は、必要な暗号化回路、復号化回路、メモリのコストを削減し、安価かつ高性能なマイクロプロセッサの提供にある。

【0010】本発明の第3の目的は、ネットワークを介して、クライアント側装置に安全にプログラムを配布することのできるプログラム配布装置の提供にある。

【0011】本発明の第4の目的は、ネットワークを介して配布されたプログラムを安全に受信することのできるクライアント側装置の提供にある。

【0012】本発明の第5の目的は、ネットワークを介して安全にプログラムを配布することのできるプログラム配布システムの提供にある。

【0013】本発明の第6の目的は、公衆ネットワークを介した安全なプログラムの配布方法の提供にある。

【0014】本発明の第7の目的は、暗号化されたプログラムを、オペレーティングシステムとの不整合なくロードすることのできるプログラムロード方法の提供にある。

【0015】本発明の第8の目的は、第三者による不正な解析から保護することを可能にした、実行ファイルを記録したコンピュータ読み取り可能な記録媒体の提供にある。

【0016】

【課題を解決するための手段】上記の第1の目的を達成するために、本発明のマイクロプロセッサは、外部へ読

み出すことのできない固有の秘密鍵を内部に保持し、この秘密鍵に対応する公開鍵であらかじめ暗号化された内容を復号化する1チップまたは1パッケージのマイクロプロセッサを提供する。このマイクロプロセッサは、仮想アドレスを物理アドレスに変換するアドレス変換手段と、このアドレス変換手段の変換規則と各々が前記仮想アドレスで指定された範囲の暗号化属性情報を有する1以上のエントリとを含むテーブルを格納する第1の記憶手段（たとえば変換索引バッファ）と、マイクロプロセッサ外部のメモリから指定された物理アドレスに対応する内容を読み出すメモリ読み出し手段と、前記テーブルのエントリに含まれる暗号化属性情報が、仮想アドレスで指定された範囲が暗号化されていることを示す場合に、外部メモリに記録された内容をメモリ読み出し手段を介して読み出し、前記固有の秘密鍵によって復号化する復号化手段と、復号化手段で復号された内容を一時的に記憶する第2の記憶手段と（たとえば一次キャッシュ）、第2の記憶手段に記憶された内容を逐次解釈する命令デコード手段とを備える。アドレス変換手段と、メモリ読み出し手段は、たとえばバスインターフェイスとして実現され得る。

【0017】復号化され、第2の記憶手段に格納された内容は、命令デコード手段以外からは読み出し禁止とされる。また、仮想アドレスで使用している外部メモリの対応する物理アドレスに対して読み出しが行われた場合には、この読み出しに対して例外処理を発生させるか、または物理アドレスに記録されている値とは異なる値を返す。このように、暗号化された内容の復号化処理は、マイクロプロセッサに固有の秘密鍵によって、完全にマイクロプロセッサ内部で行われ、かつ復号化されたデータの外部からの読み出しが防止される。すなわち、PCの所有者によるプログラムの解析、改変が効果的に防止される。また、暗号化属性を示す情報、たとえば暗号化フラグをエントリに設定することにより、既存のプログラムにソース変更や再コンパイルなどの変更を加えることなく、暗号化フラグの状態を見ることによって、暗号化されたプログラムを滞りなく実行することが可能になる。

【0018】また、暗号化プログラムの実行中の割り込みの発生を考慮して、マイクロプロセッサは、復号化された内容の実行状態を表わす値を保持するレジスタと、暗号化された内容の実行中に割り込みが発生した場合に、レジスタに記憶されている値を、ランダムな値を持つ任意の暗号化鍵で暗号化するレジスタ暗号化／復号化処理部とをさらに有する。暗号化に用いた任意の暗号化鍵自体を、マイクロプロセッサに固有の秘密鍵でさらに暗号化して、レジスタに保存する。割り込みが終了して、暗号化プログラムを続行する場合は、レジスタ値暗号化／復号化処理部は、レジスタの内容を復号化する。この構成により、不正な解析をより困難にし、安全性を

10

20

30

40

50

高めている。

【0019】本発明の第2の目的を達成するために、1チップまたは1パッケージのマイクロプロセッサは固有の秘密鍵とこれに対応する公開鍵を有し、公開鍵に代えて、任意の共通鍵を用いて暗号化された暗号化プログラムを実行する。暗号化に用いられた任意の共通鍵は、マイクロプロセッサに固有の公開鍵によってあらかじめ外部で暗号化され、マイクロプロセッサに保持される。このマイクロプロセッサは、仮想アドレスを物理アドレスに変換するアドレス変換手段と、アドレス変換手段の変換規則と公開鍵によって暗号化された共通鍵についてのキー情報とを含むキーテーブルを格納する第1の記憶手段（たとえば命令TLB）と、マイクロプロセッサ外部のメモリに対して、指定した物理アドレスに対応する内容を読み出すメモリ読み出し手段と、暗号化された共通鍵を秘密鍵で復号する第1の復号化処理手段と、暗号化されたプログラム内容を、復号化された共通鍵で復号する第2の復号化処理手段と、復号化されたプログラム内容を一時的に記憶する第2の記憶手段（たとえば一次命令キャッシュ）と、第2の記憶手段に記憶された内容を逐次解釈する命令デコード手段とを備える。

【0020】このマイクロプロセッサでは、任意の共通鍵（対象鍵）によってあらかじめ暗号化されたプログラムを復号化するには、まず、マイクロプロセッサに固有の公開鍵で暗号化された共通鍵を、対応する秘密鍵で復号化しなければならない。したがって、固有の秘密鍵を有するユーザでないと、暗号化プログラムを復号化することができない。一方、プログラム自体は、非対称鍵アルゴリズムよりもずっと簡単な対象鍵（共通鍵）アルゴリズムで暗号化されているので、共通鍵さえ復号できれば、プログラムの復号化は簡単にできる。これによりハードウェアのコストが低減され、復号化の速度が向上する。

【0021】キー情報は、キーテーブルを参照して、複数のエントリで共有される。これにより、鍵の保持に必要な高速メモリの量を低減することができる。

【0022】第3の目的を達成するために、上記の第1および第2の目的と関連して説明したマイクロプロセッサが内蔵されたクライアント装置に、ネットワークを介して実行プログラムを配布するプログラム配布装置を提供する。上述したように、マイクロプロセッサはあらかじめ固有の公開鍵と秘密鍵とを有する。このプログラム配布装置は、クライアント装置との間に第1の通信路を設定する第1通信路設定部と、前記第1の通信路を介して前記クライアント装置を使用するユーザの認証を行うユーザ認証部と、前記第1の通信路上に、クライアント装置が内蔵するマイクロプロセッサに直接連絡する第2の通信路をさらに設定する第2通信路設定部と、クライアント装置に配布すべき実行プログラムを暗号化して暗号化プログラムを作成する暗号化処理部と、暗号化した

プログラムを前記第2の通信路を介してクライアント装置のマイクロプロセッサに送信する配布送信部とを備える。

【0023】暗号化処理部は、第2の通信路を介してクライアント装置のマイクロプロセッサから送られてくる固有の公開鍵を用いてプログラムを暗号化する。あるいは、任意の暗号化鍵でプログラムを暗号化し、この暗号化鍵を、クライアント装置のマイクロプロセッサから送られてくる公開鍵で暗号化する。暗号化された暗号化鍵は、暗号化されたプログラムと共にクライアント装置に配布される。

【0024】第4の目的を達成するために、ネットワークを介してプログラム配布装置からプログラムの配布を受けるクライアント装置は、上述のあらかじめ固有の秘密鍵と公開鍵とを有するマイクロプロセッサと、前記プログラム配布装置との間に第1の通信路を設定する第1クライアント側通信路設定部と、第1の通信路を介して、前記クライアント装置を使用するユーザのユーザIDを前記プログラム配布装置に送信するユーザ認証部と、第1の通信路上に、前記マイクロプロセッサから直接プログラム配布装置に連絡する第2の通信路をさらに設定する第2クライアント側通信路設定部と、前記第2の通信路を介してプログラム配布装置に、マイクロプロセッサが固有の秘密鍵と公開鍵とを確かに保持することを証明する証明を送信する証明部と、第2の通信路を介して前記プログラム配布装置から、暗号化された実行プログラムを受信する受信部とを有する。

【0025】第5の目的を達成するために、本発明のプログラム配布システムは、ネットワークと、このネットワークに接続された上述したプログラム配布装置と、ネットワークに接続された上述したクライアント装置とを含む。

【0026】第6の目的を達成するために、本発明のプログラム配布方法は、固有の秘密鍵と、この秘密鍵に対応する固有の公開鍵とをあらかじめ有するマイクロプロセッサを内蔵するコンピュータから、プログラム配布装置に、ネットワークを介して第1の通信路を設定する。この第1の通信路上に、前記マイクロプロセッサからプログラム配布装置に直接連絡する第2の通信路をさらに設定する。第2の通信路を介して、前記マイクロプロセッサから前記プログラム配布装置に、前記固有の公開鍵を送信する。そして、ネットワークを介して送信された公開鍵を、プログラム配布装置で受信し、プログラム配布装置でプログラムを暗号化する。暗号化したプログラムを、第2の通信路を介して、コンピュータのマイクロプロセッサに直接送信する。

【0027】このようにして配布されるプログラムは、コンピュータが内蔵するマイクロプロセッサに固有の公開鍵で暗号化されているか、あるいは共通鍵で暗号化され、この共通鍵をマイクロプロセッサに固有の公開鍵で

暗号化してある。プログラムは、コンピュータを介さずに、直接マイクロプロセッサに送信され、この公開鍵と対応する固有の秘密鍵を有するマイクロプロセッサでなければ復号することができない。このような配布方法により、第三者による不正な復号を効果的に防止することができる。

【0028】第7の目的を達成するために、本発明のプログラムロード方法は、暗号化されたプログラムに対して1以上のページからなる仮想記憶領域を割り当て、前記暗号化されたプログラムを前記1以上のページに書き込んでから、前記1以上のページのそれぞれに対応するページテーブルエントリの暗号化フラグをセットする。これにより、既存のオペレーティングシステムとの不整合を生じさせることなく、暗号化されたプログラムを実装することができる。また、このようなロード方法を実行させるプログラムをあらかじめコンピュータ読み取り可能な記録媒体に記録してもよい。ここでいう記録媒体とは、たとえばフロッピーディスク、CD-ROM、MOディスクなどのコンピュータ外部のメモリ装置、半導体メモリ、磁気ディスク、光ディスク、磁気テープなどを含む。

【0029】さらに、第8の目的を達成するために、暗号化されたプログラムファイルを記録するコンピュータ読み取り可能な記録媒体を提供する。このプログラムファイルは、外部のプログラムへの直接の参照を含まない1以上の第1のプログラム領域と、外部のプログラムへの直接の参照を含む1以上の第2のプログラム領域を有する。第1プログラム領域は、あらかじめ定められた暗号化鍵で暗号化された実行ファイルを記録し、第2プログラム領域は、外部のプログラムモジュールへのジャンプ命令を並べたアドレス変換用のテーブルを平文の状態で記録する。このプログラムファイルがプログラムローダによりコンピュータのメモリ上に読み込まれるとき、外部への参照を含む第2領域は、そのコンピュータの外部プログラムの配置に基づいて適切なアドレスを参照するようにプログラムローダによって書き換えられる。このとき、第2領域が平文の状態でアドレス変換テーブルを格納しているので、暗号化された実行プログラムをロードするときにリロケーションで不整合が生じない。また、ロード時の不整合を防止すると同時に、暗号化されてデータを格納する第1領域により、プログラムの大部分を不正な解析から保護することが可能になる。本発明のその他の特徴、効果は、以下に述べる実施の形態によって、より明確になるものである。

【0030】

【発明の実施の形態】以下、図面を参照して本発明を詳細に説明する。

【0031】＜第1実施形態＞図1～図4は、本発明の第1実施形態に係るマイクロプロセッサ10を説明するための図である。図1の概略ブロック図に示すように、

マイクロプロセッサ10は、このマイクロプロセッサに固有の秘密鍵112と、この秘密鍵とペアをなす公開鍵114と、この公開鍵によりあらかじめ暗号化された実行プログラムを実行する際に秘密鍵112で復号化する復号化処理部116と、復号化されたプログラムを一時的に格納する一次命令キャッシュ118と、一次命令キャッシュ118に格納された内容を逐次解釈する命令フェッチ/デコード手段230と、バスインターフェイスユニット122と、命令TLB120とを、1チップ上または1パッケージ135内に備える。

【0032】バスインターフェイスユニット122は、仮想アドレスを物理アドレスに変換するアドレス変換手段（不図示）と、マイクロプロセッサ外部のメインメモリ203から、指定された物理アドレスの内容を読み出すメモリ読み出し手段（不図示）とを有する。また、命令TLB120は、アドレス変換手段の変換規則と、各々が仮想アドレスで指定された範囲が暗号化されているかどうかを示す暗号化フラグを有する1以上のエントリを含むページテーブルを読み込む。

【0033】マイクロプロセッサ10は、たとえばPCに組み込まれると、システムバス107を介して外部記憶装置101、PCのメインメモリ103、2次キャッシュ105に接続される。ユーザは、所望のプログラムを暗号化された状態で購入したい場合に、このマイクロプロセッサ10に固有の公開鍵を、PCを介してプログラムペンダに送信する。プログラムペンダは、送られてきた公開鍵でプログラムを暗号化して、暗号化された実行プログラムをユーザのPCに送り返す。暗号化された実行プログラムは、たとえばPCのメインメモリ103に図2に示すファイル形式でロードされ、実行する場合にだけマイクロプロセッサ10の復号化処理部116で復号化され、実行される。このとき、マイクロプロセッサ10に固有の秘密鍵112を用いなければプログラムの復号化はできない。

【0034】このマイクロプロセッサ10は仮想記憶性能を有する。したがって、図3に示すように、メインメモリ103の所定の範囲を指定する物理アドレスによって区切られたページ38-1～38-kと、これらに対応するエントリを有するページテーブル37が設定される。命令TLB120は、マイクロプロセッサが実行しようとする範囲のプログラムが書き込まれた物理ページに対応するページテーブルエントリを取り込む。各エントリには、上述したように暗号化フラグが設けられており、対応するページの内容が暗号化されている場合に暗号化フラグがセットされる。復号化処理部116は、エントリに暗号化フラグがセットされている場合にのみ、このエントリに対応するページの内容を復号化する。ページテーブルと暗号化フラグの詳細については、後述する。

【0035】暗号化プログラムの実行中は、復号化され

て一次命令キャッシュ118に格納されていたプログラムの命令は、順次、命令フェッチ／デコード部130によってデコードされ、命令プール131に供給される。命令実行切り替え部132は、一次データキャッシュ134から順次データをロードする。命令の実行が終了すると、命令実行完了部133を介して、メモリに書き込まれるべきデータは一次データキャッシュ134に書き戻される。

【0036】このような暗号化プログラムの実行中に、割り込みによって復号化が中断されることがあり得る。中断が起きると、従来のマイクロプロセッサでは、別のタスク（コンテキスト）への切り替えが発生するときに、プロセッサの内部状態をメインメモリ103のスタック上に書き出す。割り込み中のタスクの処理が終了したら、メインメモリ103に書かれた内容が復元され、実行が再開される。しかし、プロセッサの実行コンテキスト情報を調べれば、そこにはマイクロプロセッサの実行アドレスと内部状態に関する情報が含まれるため、プログラムがどのような動作をしていたかがわかってしまう。

【0037】これを防止するために、本発明のマイクロプロセッサ10は、レジスタ値暗号化／復号化処理部126を有する。レジスタ値暗号化／復号化処理部126は、暗号化プログラムの実行中に割り込みや例外が発生した場合に、実行済みの命令の状態を保持しているレジスタ124の内容を、一旦暗号化する。このときの暗号化の鍵は、マイクロプロセッサ10がランダムな値を割り当てたものである。この暗号化鍵をさらにマイクロプロセッサ10の固有の秘密鍵112で暗号化して、暗号化されたコンテキスト情報と共にスタック上に保存する。ただし、コンテキスト情報のうち、オペレーティングシステムがそのプロセスを再開するのに必要なレジスタ情報は暗号化されない。また、プログラムの再開アドレス、スタックポインタなどのシステムレジスタと、例外発生ページアドレスは暗号化されない。これら以外の汎用レジスタが暗号化される。

【0038】割り込みが終了し、リターン命令によってもとの暗号化プログラムに戻る場合は、マイクロプロセッサ10はスタック上の暗号化されたレジスタ値を一度レジスタ124に読み込む。レジスタ値暗号化／復号化処理部126は、レジスタ124上に読みこまれた暗号化値を復号化し、実行を再開する。より具体的には、マイクロプロセッサ10は、保存されたコンテキストの再開アドレスを調べる。再開アドレスが暗号化されたページであれば、レジスタ値暗号化／復号化処理部126は、秘密鍵112を用いてコンテキスト情報を復号化する。このような割り込み時、および復帰時のレジスタの暗号化、復号化は、いかなるプログラムからも知ること

【0039】なお、システムコール呼出しに使われるソ

フトウェアインタラプトの場合は、パラメータの受け渡しに汎用レジスタが使われるため、この汎用レジスタについては暗号化を行わない。すなわち、実行中の暗号化プログラム自体が発生させる割り込み（ソフトウェアインタラプトと呼ばれ、ユーザモードからOSのカーネルモードに制御を移すときに使われる割り込み処理）では、レジスタ124の暗号化は行わない。ソフトウェアインタラプトはシステムコール発行に伴って実行されるので、暗号化プログラムの作成者はあらかじめこの点に留意したうえで、セキュリティが守られるようにプログラムを作成する。

【0040】暗号化プログラムの実行が終了すると、命令TLB120に読み込まれたエントリにセットされていた暗号化フラグがクリアされ、一次命令キャッシュ118に格納された平文状態のプログラムデータは破棄される。このとき、復号化された内容がメインメモリ103に書き戻されることはない。これにより、完全にマイクロプロセッサ10の内部だけで、暗号化された実行プログラムの復号が行われ、他のソフトウェアもユーザも、復号化を知ることができない。

【0041】図2は、第1実施形態における実行プログラムのファイル形式を示し、図3は、第1実施形態のマイクロプロセッサ10における論理アドレス31と、仮想メモリを管理するページテーブル37との関係を示す。図4は、図3に示すページテーブルのうち、エントリ37-jの構成を示す。

【0042】図2に示すように、第1実施形態のプログラムファイルは、このプログラムの外部のプログラムへの直接の参照を含まない第1のプログラム領域（すなわち、.textセクション、.bssセクション、.rdataセクションなど）と、このプログラムの外部のプログラムへの直接の参照を含む第2のプログラム領域（すなわちIATセクション）とを有する。第1のプログラム領域には、データはあらかじめ暗号化鍵で暗号化された状態で記録され、第2のプログラム領域には、平文の状態でジャンプテーブルが記録される。

【0043】ファイルの先頭には、MS-DOSヘッダ11およびCOFFファイルヘッダ12があり、プログラム全体の属性などを定義する。プログラムは、.textセクション13、.bssセクション14など、いくつかのセクションに分かれており、各セクションごとに、属性を定義するセクションヘッダと、内容が含まれるセクション本体がある。通常、プログラムのコード領域は、.textセクションに格納される。ただし、図2に示すように、IAT（Import Address table）と呼ばれる外部プログラムへのジャンプ命令が格納される領域は、コード領域ではあっても、プログラム本体とは別の、IATセクションに格納される。セクションヘッダにはいくつかのフィールドがあり、その中の特性フィールド13-1が、セクションのメモリ上の配置境界や読み書き属性などを

10

20

30

40

50

指定している。

【0044】第1実施形態では、特性フィールド13-1上の1ビット、たとえば0x00000400でマスクされる1ビットが、暗号化属性を指定する暗号化フラグとして用いられる。このビットが1であれば、対応のセクションが暗号化されていることを示し、0であれば暗号化されていないことを示す。図2の例では、このビットが値1を有するので、対応のセクションは暗号化されていることになる。たとえば、.textセクション17の内容は、本来平文であったプログラムをマイクロプロセッサ10の固有の公開鍵114で暗号化されたものである。一方、ジャンプテーブルが格納される.IATセクションヘッダ16の暗号化フラグは値0を持ち、このセクションにはジャンプテーブルが暗号化されず、平文のまま格納されることを示す。

【0045】一般に、プログラムがロードされるアドレスは、システム構成や、そのメモリの利用状況によって異なる。そのようなとき、プログラムに含まれる参照アドレスをプログラムローダが状況に合わせて変更する必要が生じる。これをリロケーションと呼ぶ。あるプログラム内部に閉じた関数呼出しは、プログラム内部の相対的なアドレスに基づいてアクセスを行うリロケータブルなコードを生成することによって、リロケーション処理を不要にすることができる。しかし、プログラム外部のモジュールの呼出し、たとえばシステムコール呼出しや、その逆に外部のモジュールからプログラム内部の関数が呼び出される場合には、モジュールの絶対アドレスも相対的な位置関係も、プログラムがロードされるまでわからないことが多い。システムのハードウェア構成やソフトウェアのバージョンによってプログラムや作業領域のサイズが変わることがその理由である。

【0046】この問題を解決するため、次のような手法を用いる。外部関数への呼出しを含むプログラムファイルに、IAT (Import Address Table) と呼ばれる、読み出される外部関数へのジャンプ命令を並べたテーブルをあらかじめ作成しておき、暗号化されたプログラム本体（ここでは、.textセクション17）から外部モジュールへの呼出しは、このテーブルへのコール命令として間接的に行う。ジャンプテーブルには、呼出し先の関数を識別する名前が付加されており、プログラムローダはプログラムのロード時に名前に基づいて外部関数のアドレスを検索し、上記テーブルの対応する命令のジャンプ先をその関数のアドレスに書き換える。この状態でプログラムから外部関数への呼出しが行われると、制御は一度IATのジャンプ命令に移り、次にジャンプ命令の飛び先の目的とする関数に制御が移る。第1実施形態では、.IATセクション20にこのジャンプテーブルが格納されている。

【0047】IAT領域はプログラムローダによる書き換えが行われるので、この領域が暗号化されていると、

書き換えが正常に行われなくなり、正しい外部関数が呼び出されないことになる。そこで本発明では、図2に示すように、IATをプログラム本体(.textセクション17)とは別のセクションに設け、.IATセクション20を平文のまま維持することにより、リロケーション処理が正しく行われる。IATに含まれる情報は、外部関数への呼出しだけなので、プログラム自体の秘密が損なわれることはない。

【0048】第1実施形態では、IATを外部関数へのジャンプ命令が格納されていることとしたが、プログラム本体からの呼出し命令が、IATの内容で指定されるような間接呼出し命令を使っている場合には、IATの内容はジャンプ命令ではなく、単なるとび先のアドレスが格納されたデータ列であってもよい。もちろんこの場合も、IATは暗号化されない。

【0049】次に、ページテーブルエントリに設けられた暗号化フラグについて説明する。暗号化された実行プログラムは、ローダプログラムによってメインメモリ103上に読み込まれる。このとき、ローダは、ロードするプログラムのために1以上のページから成る仮想メモリ空間を割り当て、これらのページにプログラムのすべてのセクションを書き込んで配置する。配置が終わると、ローダは、.textセクション17が配置されたページテーブルエントリ37-jの暗号化フラグ37-j-E（図4）を1にセットする。このページテーブルエントリ37-jが、マイクロプロセッサ10の命令TLB120にキャッシュされているときに暗号化フラグがセットされていると、バスインターフェイスユニット121がそれを監視していて、一次命令キャッシュ118と二次命令キャッシュ105の、このページエントリに対応するキャッシュエントリを無効化して、復号化された内容を記憶するためのスペースをあける。このとき、平文状態の.IATセクションが配置されたページテーブルエントリの暗号化フラグは0のままである。

【0050】いったん、ページテーブル307-j-Eの暗号化フラグがセットされると、以後このページテーブルに対応する物理メモリのページ38-kへの他プログラムからのアクセスは禁止される。すなわち、このページへの読み出し、書き込みは一切できなくなる。この保護は、実行がユーザモード、カーネルモードのどちらの場合にも適用される。

【0051】暗号化フラグがセットされている間に、外部から読み書きの要求があった場合は、保護例外が発生する。あるいは、読み出しに対しては、常に所定の値（たとえば0）もしくは乱数が読み出され、書き込みに対しては無効とするようにしてもよい。例外の扱いについては後述する。暗号化フラグがセットされた物理ページ310-kが平文状態で一次データキャッシュ134にキャッシュされることはない。以下、ページテーブルの暗号化フラグがセットされたページに対応するメモリ領域

を暗号化ページまたは暗号化領域と呼ぶ。暗号化領域の実行では、プロセッサのデバッグ機能、たとえばステップ実行機能などは無効化される。

【0052】ローダは、メインメモリ103上にページテーブルを設定すると、プログラムのエントリポイントに制御を移し、プログラムの実行が開始される。マイクロプロセッサ10が命令として暗号化領域にアクセスすると、キャッシュのミスヒットが生じ、メインメモリ103から一次命令キャッシュ118への読み込みが始まる。このとき、暗号化領域については、復号化処理部116が秘密鍵112を使用して内容を復号化し、一次命令キャッシュ118上には、実行可能な命令列が読み込まれ、マイクロプロセッサ10はそれを実行する。暗号化領域以外をキャッシュする場合には、復号化処理部116はなににもせずに、メインメモリ103の内容はそのまま一次命令キャッシュ118にコピーされる。

【0053】上述したように、この復号化はすべてマイクロプロセッサ10内部のハードウェアである一次命令キャッシュ118の動作の一部として行われる。したがって、前述したページテーブルの扱いを除けば、ソフトウェアは復号化操作を意識することはまったくない。ページテーブルの設定についても、通常はオペレーティングシステムによって設定が行われるので、アプリケーションからはまったく見ることができない。さらに、これも上述したように、マイクロプロセッサ10に固有の秘密鍵112は、マイクロプロセッサ10の内部に隠され、プログラムからはその値を読み出すことはできない。したがって、暗号化された実行プログラムの内容は、マイクロプロセッサ10が組み込まれたPCのアプリケーションプログラムはもちろん、オペレーティングシステムからも復号化された形で読み出すことは不可能である。

【0054】(ページの無効化) プログラムの実行が終了すると、オペレーティングシステムは他のプログラムを動作させるために、そのプログラムの.textセクションが読みこまれていた暗号化領域のメモリページを開放して、別の内容の書き込みを認める必要がある。ページテーブルエントリに暗号化フラグがセットされたままでは、対応するページへの読み書きはできないので、ページテーブル上の暗号化フラグをリセットして、読み書きを許可する。仮想記憶を採用している場合は、プログラムの実行中にも、ページの一部がページアウトされて、ページの内容を二次メモリに書き出して、物理メモリを他の目的に使う必要が生じることがある。この場合も、ページの内容を読み出す前に、暗号化フラグはリセットされる。暗号化フラグがリセットされると、それを監視するバスインターフェイスユニット122は一次命令キャッシュ118の内容を無効化する。一次データキャッシュ134については、暗号化フラグがセットされた状態で一次データキャッシュ134への暗号化領域の読み

込みが禁止されていれば、無効化の必要は特でない。

【0055】(メモリマップトファイル) オペレーティングシステムの仮想記憶の実装によっては、プログラムの開始時にすべてのイメージを実際の物理メモリが読み込まれない場合もある。すなわち、プログラムの実行開始時にプログラムのイメージが仮想アドレスに割り付けられるだけで、物理メモリにロードされない場合もある。この仮想アドレスへの割付は、メモリマップトファイルと呼ばれる。通常、ページテーブルには、そのページが物理メモリに存在するかどうかを示すフラグがある。図4では、右下の37-j-pがこのフラグに相当する。

【0056】通常、仮想メモリのあるページがアクセスされたとき、その仮想ページに物理メモリが割り当てられていない場合、オペレーティングシステムは例外原因を調べて、仮想ページに対応する物理ページを確保してからページテーブルを更新する。そして、ページ内容を二次メモリから読み出して、確保した物理メモリに書き込んでから、実行を再開する。対応のページが物理メモリに存在せず、かつ暗号化フラグがセットされている場合には、オペレーティングシステムは、まず物理メモリのスペースを確保してページテーブルを更新した後、ページ内容を書き戻すために、一度暗号化フラグをクリアしてから、通常の場合と同様に、二次メモリの内容を物理メモリに書き込み、その後再度暗号化フラグをセットして暗号化プログラムの実行を再開する。

【0057】(デバッグフラグ) デバッグのためにプログラムの動作を検証するときには、そのプログラムが実行される実際の環境にできる限り近いことが望ましい。その意味では、暗号化プログラムのデバッグは、暗号化状態で行うべきである。しかし、プログラムが暗号化された保護状態では、ステップ実行などのデバッグ機能が使えず、コードを逆アセンブリすることもできないため、デバッグは事実上不可能に近い。このため、暗号化プログラムのデバッグは、どうしても平文状態で行う必要があると考えられる。

【0058】暗号化状態と平文状態での実行の差について考えると、マイクロプロセッサ10でのコードの実行については、マイクロプロセッサ10の内部では、プログラムが復号化された状態で実行されるため、暗号化された状態と平文の状態とで、動作に本質的な違いはない。しかし、メモリ保護については、暗号化されたプログラムを実行する場合、復号化されたプログラムを保護するため暗号化されたページへの読み出しを禁止している点で動作が異なっている。

【0059】そこで、暗号化プログラムのデバッグを、デバッグ作業の容易な平文状態で行い、平文状態で行い、かつデバッグ動作を暗号化状態と一致させるため、ページテーブルにデバッグフラグを設けている。図4に示すページエントリでは、37-j-Dがデバッグフラグに相当する。デバッグフラグがセットされているときに、そ

のページに対して暗号化プログラムの実行以外の読み書きが行われると、暗号化プログラムの保護例外が発生する。しかし、デバッグ作業は平文状態で行われているので、実行コンテキストは平文のままセーブされ、ステップ実行も可能となる。すなわち、プログラム実行においては、デバッグは平文のプログラムであっても、振る舞いは暗号化された状態と一致する。かつ、ステップ実行による実行トレースが可能になる。また、保護による例外が発生した場合も、セーブされた実行コンテキストを解析することにより、例外の発生原因を調べた上で、実行を再開することもできる。

【0060】ページテーブルについては、暗号化プログラムと平文のプログラムの間で暗号化フラグの値の相違は残るが、一般にアプリケーションプログラムはページテーブルの内容を意識することはないので、アプリケーションプログラムのデバッグの妨げにはならない。

【0061】暗号化プログラムのセキュリティを守るため、暗号化フラグがセットされた状態では、デバッグフラグをセットしても、ステップ実行などは禁止されたままにする。または、暗号化フラグがセットされた状態では、デバッグフラグをセットできないようにしてもよい。なお、暗号化フラグがセットされている限り、デバッグフラグの状態に関わらず、暗号化領域からのデータ読み出しは禁止される。

【0062】＜第2実施形態＞図5～図8は、本発明の第2実施形態に係るマイクロプロセッサ20を説明するための図である。第1実施形態では、マイクロプロセッサに固有の公開鍵を用いて暗号化されたプログラムを実行していた。しかし、公開鍵方式の暗号化／復号化アルゴリズムは、共通鍵方式に比較して一般に複雑で、回路規模が大きくなり、コストが高くなる上に、処理の高速化が困難である。そこで、第2実施形態のマイクロプロセッサは、共通鍵で暗号化されたプログラムを復号し実行するとともに、共通鍵を安全かつ効率的に管理する。

【0063】具体的には、マイクロプロセッサに供給される実行プログラムは、プログラムベンダが選択した任意の暗号化鍵（すなわち共通鍵）で暗号化される。暗号化に用いた共通鍵は、マイクロプロセッサに固有の公開鍵で暗号化される。共通鍵で暗号化された暗号化プログラムと、公開鍵によって暗号化された共通鍵とがマイクロプロセッサに送られてくるので、マイクロプロセッサは、固有の秘密鍵で共通鍵を復号化し、復号化された共通鍵で、暗号化プログラムの内容を復号化する。これにより、セキュリティを維持したまま、復号化のためのハードウェアコストを低減することができる。

【0064】図5に示すように、マイクロプロセッサ20は、プログラムから読み出すことのできない秘密鍵と212、この秘密鍵に対応する公開鍵214と、この公開鍵であらかじめ暗号化された状態で外部のメモリに格納されている第3の暗号化鍵E_{Kp} [K_x]を前記秘密

鍵212を使用して復号化する復号化鍵TLB（第1の復号化処理手段）236と、外部のメモリに格納されている暗号化されたプログラムを、前記復号化された第3の暗号化鍵E_{Kp} [K_x]によって復号化する復号化処理部216（第2の復号化処理手段）と、復号化されたプログラムを格納する一次命令キャッシュ218と、バスインターフェイスユニット222と、命令TLB220とを、1つのチップまたはパッケージ内に備える。

【0065】バスインターフェイスユニット222は、仮想アドレスを物理アドレスに変換するアドレス変換手段（不図示）と、マイクロプロセッサ外部のメインメモリ203から、指定された物理アドレスに対応する内容を読み出すメモリ読み出し手段（不図示）とを有する。命令TLB220は、アドレス変換手段による仮想アドレスから物理アドレスへの変換規則と、公開鍵によってあらかじめ外部で暗号化された第3の暗号化鍵E [K_x]（共通鍵）についてのキー情報とを含むテーブルを取り込む。より具体的には、命令TLB220は、まずメインメモリ203のアドレスによって区切られたページ領域の各ページに対応するエントリを含むページテーブルを取り込む。各エントリは、対応するページの暗号化属性を示す暗号化フラグと、このページの内容を復号するための鍵情報が記憶されているキーテーブル上の位置を指定する識別子（キーエントリID）とを有する。この識別子を参照することによって、対応するキーテーブルのエントリを取り込む。

【0066】図6は、第2実施形態のマイクロプロセッサ20で実行する暗号化プログラムを格納するファイル形式を示す。textセクションヘッダ63のフィールドには、暗号化アルゴリズム63-2のフィールドと、暗号化鍵値63-3のフィールドが追加されている。暗号化アルゴリズム63-2のフィールドには、コード領域を暗号化するための暗号化アルゴリズムが符号化されて格納される。暗号化鍵値63-3のフィールドには、マイクロプロセッサ20が暗号化プログラムを解読するために使う共通鍵K_xが、マイクロプロセッサ20に固有の公開鍵214で暗号化された値E_{Kp} [K_x]が格納されている。共通鍵K_xの長さは、たとえば、暗号にトリプルDESを使った場合には192ビットが必要になる。そして、E_{Kp} [K_x]は、マイクロプロセッサ20の公開鍵214によって暗号化される際のブロック長の整数倍のサイズとなる。

【0067】textセクション66は、共通鍵K_xを用い、暗号化アルゴリズム63-2のフィールドで指定されたアルゴリズムによって暗号化されている。ここでは、暗号化アルゴリズムには共通鍵方式のトリプルDESが指定されているものとする。特性フィールド63-1には、プログラムが暗号化されていることを示すため、暗号化フラグが1にセットされていることは第1実施形態と同様である。

【0068】プログラムベンダが図2に示すような暗号化ファイルを作成するには、まず、コード領域の暗号化鍵を共通鍵Kxとしてランダムに選び、.textセクション66を、64ビット（8バイト）ごとのブロック単位でトリプルDESアルゴリズムで暗号化する。暗号化鍵値63-3には、解読用共通鍵Kxをターゲットであるマイクロプロセッサ20の公開鍵214で暗号化した値EKp[Kx]を格納し、暗号化アルゴリズムフィールド63-2にはトリプルDESの符号表現を入れ、特性フィールドの暗号化フラグを1にセットする。

【0069】なお、コードを格納する領域を.textセクション66に限らず複数のセクションとすることにより、同一プログラムのコードを複数の暗号化鍵もしくは暗号化アルゴリズムで暗号化してもよい。ただし、それぞれのセクションはプロセッサの仮想記憶（図7）のページ境界ごとに配置されなければならない。

【0070】また、あらかじめ、プログラムのコード領域の一部に、プログラムの動作と関係のない部分を設け、コード領域を暗号化する前に、その部分にプログラムを使用するターゲットのマイクロプロセッサ20の公開鍵214やプログラムの配布先ユーザIDを直接、あるいは暗号化した形式で格納してもよい。こうすることによって、プログラムが不正解読され、不正コピーが配布された場合に、その経路を追跡することが可能になる。経路追跡のための情報は、プログラムの1サブルーチンとして実行可能な機械語命令列に符号化され、発見を困難にすることも可能である。

【0071】さて、暗号化されたプログラムをマイクロプロセッサ20で実行する際には、実施形態1と同様に、まず、マイクロプロセッサ20が組み込まれたPCのメインメモリ203に、暗号化プログラムをローダによって読み込むことから始まる。ローダは、暗号化プログラムのための仮想メモリ空間を含むプロセスコンテキストを設定し、すべてのセクションを通常のプログラムを同様にメモリ上に配置し、必要に応じてリロケーションなどを行う。

【0072】図7は、第2実施形態に係るマイクロプロセッサ20で暗号化プログラムを実行する際のページテーブルを示す。図7に示す第2実施形態のページテーブルはキーテーブル79を有する拡張テーブルである。論理アドレス71は、ディレクトリ72、テーブル73、オフセット74の3つのフィールドに分割されている。論理アドレス71を構成するこれら3つのフィールドは、次のようにして物理アドレスを計算する。

【0073】まず、最上位のディレクトリフィールドは、ディレクトリテーブル76のエントリ76-iを指定する。ディレクトリテーブル76の先頭は、マイクロプロセッサ20のレジスタ75によって指定される。ディレクトリエントリ76-iは、ページテーブル77の先頭へのポインタを持つ。論理アドレス71のテーブル73によ

って、ページテーブル77での位置が計算され、対応するページエントリ77-iにそのアドレスが示す実際の物理アドレス80-kが記述されている。ここまでは、第1実施形態と同様である。

【0074】図8は、図7に示したページテーブル77のエントリと、対応するキーテーブル79のエントリを示す。第2実施形態では、ページテーブル77の各エントリに、対応するページを復号化するための鍵情報を指定する番号（識別子またはキーエントリID）を格納するフィールド77-j-kが設けられる。この番号は、キーテーブル79上の、復号化鍵が格納されているエントリの番号を指定する。キーテーブル79の先頭はマイクロプロセッサ20の制御レジスタのひとつであるキーテーブル制御レジスタ78によって指定される。なお、各エントリは、第1実施例と同様に、対応するページの属性を示す暗号化フラグ77-j-eと、デバッグフラグ77-j-dも有する。

【0075】次に、ローダは暗号化されたプログラムを復号化するための共通鍵Kxをマイクロプロセッサ20に固有の公開鍵214で暗号化した値EKp[Kx]を格納するためのエントリ79-mを、キーテーブル79の上に確保する。キーテーブル79には、鍵情報と暗号化アルゴリズムのフィールドに加えて、参照カウンタのフィールド79-m-1があり、そのテーブルを参照するテーブルの個数が書き込まれる。すなわち、参照数が0であれば、そのテーブルは使用可能となる。ローダは使用するテーブルを決定して、キーテーブルの参照カウンタを0から1に書き換えてから、鍵情報と暗号化アルゴリズム情報をキーテーブルのフィールド79-m-3と79-m-4のそれぞれに書き込み、そのキーエントリの番号を対応するページテーブル77のエントリ77-jに書き込む。同じキーエントリを参照するページが複数あれば、キーテーブル79の参照カウンタを増やしてからページテーブル77のエントリにキーエントリの番号を書き込む。

【0076】すでに同一のプログラムが実行されているなどの理由により、同一の鍵がキーテーブル79に格納されている場合は、テーブルの確保は行わず、キーテーブルの参照カウンタを増やしてから、ページエントリにキーエントリの番号を書き込む。このようにして、暗号化鍵E[Kx]を複数ページで参照できるようにする。

【0077】上述のように、独立のキーテーブル79を設けて複数のページで共通の復号鍵情報を共有できるようにした理由は、ページテーブルに必要なメモリの量を減らして命令TLB220の効率を改善するためである。すなわち、第1実施形態では、暗号化プログラムの中のすべてのプログラムについて同一の公開鍵114が用いられていたため、各ページテーブルごとに鍵情報を格納する必要はなかったが、第2実施形態ではプログラムごとに必要な共通鍵の鍵情報が異なるため、ページごとに別々の鍵を格納する必要が生じる。しかし、比較的

長いビット長を要する鍵情報を、直接ページテーブルのフィールドに割り付けると、ページテーブルの大きさが増大してメモリ203の利用効率が低下する。たとえば、ページが4 Kバイトのときに、キャッシュラインサイズと同一の32バイトの鍵をページテーブルの各ページに割り当てると、ページサイズの1%弱のメモリがページテーブルのために必要となってしまう。そこで、ページテーブルの大きさを増大させたとしても、命令TLB220のサイズが以前と同じであれば、命令参照に対する命令TLB220のヒット数が低下し、プログラムの実行速度が低下する。命令TLB220のサイズも同時に大きくすることも考えられるが、高速な論理-物理アドレス変換が要求される命令TLB220の性質上、コスト高になってしまう。

【0078】また、実際には、暗号化鍵の種類は必ずしもページごとに変更する必要はなく、プログラムの秘密保護のためには、ただか数種類の鍵が使い分けられれば十分であることが多い。そこで、第2実施形態では、複数のページが同一の復号化鍵情報を共有できるように独立のキーテーブル79を設け、ページテーブル77にはキーテーブル79への番号を格納することにより、ページテーブルにル77に必要なメモリ量を低減させたのである。

【0079】さて、プログラムの配置とページテーブルの鍵情報の設定が終わると、ローダは、textセクションヘッダ63が配置されたページテーブルの暗号化フラグを1にセットする。そのページテーブルが命令TLB220にキャッシュされている場合にページテーブルの暗号化フラグが1にセットされると、バスインターフェイスユニット222がそれを監視していて、一次命令キャッシュ218および二次命令キャッシュ205の対応するエントリを無効化する。

【0080】上述のキーテーブル79の値は復号化鍵TLB26にも格納される。キーテーブル79が書き換えられた場合は、対応する復号化鍵TLB236のエントリも書き換えられる。ここで、復号化鍵TLB220には、メモリ上の復号化鍵情報E[Kx]が、マイクロプロセッサ20に固有の公開鍵214で復号化されて格納される。このとき、メモリ203上の復号化鍵情報E

[Kx]の長さを、復号化鍵TLB220上の復号化された鍵情報Kxより長く取ることにより、鍵の安全性とTLB220の利用効率を両立させることができる。マイクロプロセッサ20内部の復号化鍵TLB220に格納された鍵情報は、ユーザプログラムからは見ることができないからである。もちろん、ユーザプログラムによるキーテーブルの明示的なメモリ参照に対しては、復号化された値Kxではなく、もとの暗号化されたままの値E[Kx]が返される。

【0081】キーテーブル79の設定が完了すると、第1実施形態と同様に、ローダはプログラムのエントリポ

イントに制御を移し、プログラムの実行が開始される。プロセッサが命令として暗号化領域にアクセスすると、キャッシュのミスヒットが生じ、メインメモリ203から一次命令キャッシュ218への読み込みが始まる。このとき、ページテーブル77からキーテーブル79への参照で指定されたキーテーブル79のエントリが、マイクロプロセッサ20の秘密鍵212で復号化されて、復号化TLB236に読み込まれる。復号化された鍵情報Kxを使って、メインメモリ203に格納されていた暗号化データが復号化処理部216によって復号化され、一次命令キャッシュ218へと読み込まれ、実行される。

【0082】以後に行われる実行終了後のキャッシュの無効化、割り込み処理などは第1実施形態と同様である。

【0083】このように、第2実施形態ではプログラムの復号化に共通鍵方式を採用することができる。共通鍵のアルゴリズムは一般に同一鍵長さの公開鍵アルゴリズムと比較してハードウェア化が容易であり、復号化機能のコストを低減することができる。さらに、共通鍵をマイクロプロセッサに固有の公開鍵で暗号化し、その復号はマイクロプロセッサ20に固有の秘密鍵でないとできないので、安全性も確保される。

【0084】＜第3実施形態＞図9は、本発明の第3実施形態にかかるプログラム配布システム90の図である。このプログラム配布システムは、基本的に、第1または第2の実施形態で説明したマイクロプロセッサを内蔵したコンピュータシステムにプログラムを配布するように設計されている。したがって、配布される実行プログラムは、マイクロプロセッサ固有の鍵で直接暗号化されるか、あるいは任意の共通鍵で暗号化され、共通鍵そのものをマイクロプロセッサ固有の鍵で暗号化して実行プログラムに添付する。このため、同一のプログラム媒体のコピー、たとえば大量にプレスされるCD-ROMなどによってプログラムを配布することができない。そこで、ネットワークを介してプログラムを配布する。

【0085】図9に示すプログラム配布システム90は、ネットワーク95と、ネットワーク95に接続され、実行プログラムをネットワーク95を介して配布するプログラム配布装置93と、同じくネットワーク95に接続され、ネットワークを介してプログラム配布装置93から実行プログラムの配布を受けるクライアント装置91とを含む。

【0086】プログラム配置装置93は、クライアント装置91との間に第1の通信路を設定する第1通信路設定部931と、記第1の通信路を介してクライアント装置91を使用するユーザのユーザ認証を行うユーザ認証部933と、第1の通信路上に、クライアント装置が内蔵するマイクロプロセッサに直接連絡する第2の通信路を設定する第2通信路設定部934と、第2の通信路を

10

20

30

40

50

介して、クライアント装置91のマイクロプロセッサ901が正当であることを認証するプロセッサ認証部935と、実行プログラムをプロセッサ901と対応付けられた暗号化鍵で暗号化して暗号化プログラムを作成するプログラム暗号化処理部936と、暗号化したプログラムを第2の通信路を介してクライアント装置に配布するネットワークインターフェイス931とを備える。

【0087】クライアント装置91は、あらかじめ固有の秘密鍵と公開鍵とを有するマイクロプロセッサ901と、プログラム配布装置93との間に第1の通信路を設定する第1クライアント側通信路設定部905と、第1の通信路を介してこのクライアント装置を使用するユーザのユーザIDを送信するユーザ認証部910と、第1の通信路上に、マイクロプロセッサから直接プログラム配布装置に連絡する第2の通信路を設定する第2クライアント側通信路設定部906と、第2の通信路を介して、このマイクロプロセッサが確かにこのマイクロプロセッサだけに固有の公開鍵と秘密鍵を有することを証明する証明をプログラム配布装置に送信する証明部907と、暗号化された実行プログラムを第2の通信路を介して受信するプログラム受信部908とを備える。

【0088】第2の通信路は、第1の通信路と同じ回線の異なるチャンネル上に設定されてもよいし、第1の通信路と異なる回線上に設定されてもよい。暗号化されたプログラムは、マイクロプロセッサの認証が行われた上で、マイクロプロセッサに直接接続された第2の通信路を介してマイクロプロセッサに送信されるので、プログラム配布の安全性が確保されるとともに、ユーザによる不正を防止することができる。

【0089】図10は、図9のプログラム配布システム90におけるプログラム配布のシーケンス図である。

【0090】プログラム配布装置（サーバ）93は、全世界一意となるような公開鍵と秘密鍵のペア $K'p$ 、 $K's$ と、配布プログラムの原型である暗号化されていないプログラムファイルを有する。一方、クライアント装置91はマイクロプロセッサ901を有し、マイクロプロセッサ901には、ネットワークを介してプログラムをダウンロードするための専用プログラムが添付されている。ダウンロードプログラムは、マイクロプロセッサ901の購入時、あるいはマイクロプロセッサ901を内蔵するシステムの購入時に添付されるか、あるいは郵送などのユーザ個別に配布が可能な任意の手段でよい。

【0091】ダウンロードプログラムには、マイクロプロセッサ901ごとに異なる固有の公開鍵 Kp 、証明書などの情報が埋め込まれている。具体的には、マイクロプロセッサ901の公開鍵 Kp が実在するマイクロプロセッサに対応するものであることを示す証明書 $Cert$ が、ダウンロードプログラムに格納されている。証明書を待つ目的は、ユーザが実際のマイクロプロセッサ901と対応しない既知の公開鍵と秘密鍵の組み合わせを使って

暗号化されたプログラムを取得し、既知の秘密鍵で復号化することで平文プログラム内容を取得することを防ぐためである。証明書には、公開鍵 Kp と、 Kp を証明機関の鍵 $Kcert$ で署名した値 $SKcert[Kp]$ があらかじめ計算され、含まれている。 $Kcert$ の値は認証機関によって秘密に管理され、ダウンロードプログラムには含まれない。証明書に含まれる公開鍵 Kp と署名 $SKcert[Kp]$ を、認証用の公開鍵 $Kval$ で検証することにより、 Kp が認証機関により正しく割り当てられたものであることが証明され、既知の鍵の組み合わせを使用できないようにしてある。

【0092】このような証明書は、マイクロプロセッサにあらかじめ組み込まれていてもよい。しかし、証明書に有効期限を設けた上でダウンロードプログラムに格納し、マイクロプロセッサに添付して販売するほうが、安全性がより高められる。この場合、定期的に更新された証明書を格納したダウンロードプログラムが、マイクロプロセッサの購買者に配布されることになる。

【0093】ダウンロード過程で使用される秘密鍵 Ks は、ダウンロードプログラムのデータ領域に暗号化されて格納されている。暗号化された秘密鍵 Ks の復号に必要な鍵は、定数としてダウンロードプログラムのコード中に分散して埋め込まれている。ダウンロードプログラムが証明の過程で使用する秘密鍵 Ks は、マイクロプロセッサ901が固有に有する秘密鍵 Ks と同じ値ではあるが、マイクロプロセッサ901から読み出して得られるのではない。マイクロプロセッサが固有に有する秘密鍵 Ks がプログラムによって読み出されることは決してないのである。

【0094】一般に、あるホストが特定の公開鍵を持つことを証明するには、認証局CA（Certificate Authority）が使われることが多い。本システムにCAを使う場合は、マイクロプロセッサ901の購入時に、マイクロプロセッサ901の公開鍵とユーザ（購入者）との組み合わせをCAに登録するか、オンラインでマイクロプロセッサ901の公開鍵と利用者の組み合わせを登録する。この場合は、図10に示すプログラム配布装置（以下、「サーバ」とする）93により証明書の確認を省略することができる。

【0095】しかし、認証局CAを利用するには、ユーザが購入したマイクロプロセッサ901を認証局に登録しなければならないという不便がある。そこで、第3実施形態では、サーバ93とクライアント装置91との間だけで認証が行えるように、暗号化された証明書をダウンロードプログラムに格納することとしている。なお、第3実施形態では、サーバ93は証明書を盗用するなどの不正がいっさいない信用できるものであると仮定し、サーバ93が全世界一意の鍵のペア $K'p$ と $K's$ を持つことは、認証局などによってあらかじめ確認されているものとする。

【0096】図10のシーケンスにおいて、まずシーケンス1001において、クライアント装置91とサーバ93との間に、それぞれの装置の第1通信路設定部により、安全な第1の通信路を設定する。具体的には、クライアント装置91から、ネットワーク95を介してサーバ93に通信開始要求を送り、通信路の秘密を守るための鍵共有を行う。これは周知の鍵共有プロトコル、DH方式などでよい。以後のクライアント装置91とサーバ93との間の通信は、こうしてネットワーク95に設定された、盗聴に対して安全な通信路を通じて行う。

【0097】第1通信路が設定されたら、シーケンス1002で、クライアント装置91はサーバ93に対してダウンロードしたいファイル（プログラム）を要求し、サーバ93はクライアント装置91とユーザレベルの認証や課金処理などを行う。ダウンロードの過程で、処理の秘密をユーザから守るため、以下に説明するクライアント装置91でのダウンロードシーケンスの少なくとも一部は、暗号化コードとして実行される。ここでは、暗号化コードによって実行される部分を、マイクロプロセッサ901の動作として表現する。

【0098】シーケンス1003では、第1通信路上に、クライアント装置91のマイクロプロセッサ901とサーバ93とを直接接続する、安全な第2の通信路を設定する。

【0099】本発明では、ユーザがダウンロード過程で処理されるデータの一部を不正に取得することを防止するため、ダウンロードプログラムでは、コード自身はもちろんのこと、処理過程でメモリ上に置かれるデータについても、ユーザが読み取り理解することが困難となるように記述されている。この目的をさらに強化するため、シーケンス1003では、サーバ93とマイクロプロセッサ901との間でも秘密鍵による通信の暗号化を行う。

【0100】秘密鍵の共有を行わない場合、ユーザによる不正、たとえば、サーバ93とマイクロプロセッサ901との間の通信メッセージで、マイクロプロセッサ901の持つ公開鍵 K_p を偽の公開鍵にすりかえて暗号化されたプログラムを取得し、偽の公開鍵に対応する既知の秘密鍵で復号化することによって平文のプログラムがユーザの手にわたるおそれがあるからである。サーバ93とマイクロプロセッサ901との間の通信を秘密鍵で暗号化することにより、上述のようなユーザの不正を防ぐことができる。以後、「マイクロプロセッサ901とサーバ93との間の通信」という場合、マイクロプロセッサ901上の耐タンパなプログラムとサーバ93とが共有する暗号鍵によって暗号化され、保護された通信を意味する。

【0101】安全な第2通信路の設定後、マイクロプロセッサ901とサーバ93は相互認証を行う。すなわち、シーケンス1004で、マイクロプロセッサ901

はサーバ93に対するチャレンジのための乱数 R_c を生成し、マイクロプロセッサ901に固有の公開鍵 K_p とともに、第2通信路を介してサーバ93に送信する。チャレンジを受け取ったサーバ93は、シーケンス1005で、サーバ93の秘密鍵 K_s で乱数 R_c を暗号化した署名 $SK'_s[R_c]$ を、サーバ93からのチャレンジ R_s およびサーバ側公開鍵 K'_p とともに、マイクロプロセッサ901に送信する。（図10では、署名 $SK'_s[R_c]$ を $S[R_c](K'_s)$ と表記し、類似の表現も同様の表記とする。）

マイクロプロセッサ901は、サーバ93から送られてきた署名 $SK'_s[R_c]$ が、サーバの公開鍵 K'_p でハッシュ化された $V_{K'_p}[R_c]$ と一致するかどうかを確認する。不一致の場合は、サーバ93の認証に失敗して、以後の処理を中止する。認証に成功すれば、シーケンス1006でサーバ93のチャレンジに対して $SK_s[R_s]$ を計算し、証明書 $Cert$ をサーバ93に送る。サーバ93は、レスポンス $SK_s[R_s]$ を $V_{K_p}[R_s]$ と比較して、一致しなければ処理を中止する。一致した場合は、マイクロプロセッサ901の公開鍵 K_p と認証機関の公開鍵 K_{val} とから、 $V_{K_{val}}[K_p]$ を計算し、証明書 $Cert$ から公開鍵に対する署名 $S_{K_{cert}}[K_p]$ と照合し、検証に失敗すれば処理を中止する。

【0102】照合できた場合は、 $E_{K'_p}[Cert]$ を復号化して $Cert$ を取り出し、検証する。ここでも検証に失敗すれば処理を中止する。検証に成功して、証明書によりマイクロプロセッサ901が公開鍵 K_p を持つことを確認すると、サーバ93は実行プログラムをマイクロプロセッサ901の公開鍵 K_p で暗号化した暗号化プログラム $E_{K_p}[Prog]$ を作成する。このとき、第1実施形態で説明したように、プログラムのコード部分をマイクロプロセッサ901の公開鍵 K_p で暗号化する。このとき、第1実施形態で説明したように、プログラムのコード部分をマイクロプロセッサ901の公開鍵 K_p で暗号化する。暗号化にあたっては、第1実施形態で説明したように、プログラム本体の $text$ セクションは暗号化するが、ジャンプテーブルの IAT セクションは平文のままとしておく。

【0103】シーケンス1007で、サーバ93は暗号化プログラム $E_{K_p}[Prog]$ と、サーバ93自身の秘密鍵 K'_s による署名 $SK'_s[E_{K_p}[Prog]]$ とを、第2通信路を介してマイクロプロセッサ901に送る。この暗号化プログラムと署名は、直接マイクロプロセッサ901とサーバ93との間に確立された第2通信路を介して送られるため、クライアント装置91はこれを傍受することはできない。

【0104】マイクロプロセッサ901はプログラムの受信を完了すると、シーケンス1008でクライアント装置91にダウンロード終了を通知する。シーケンス1

10

20

30

40

50

009で、クライアント装置91はサーバ93に課金処理を要求し、サーバ93は領収書Rcptと、領収書のサーバ93の秘密鍵による署名SK' s [Rcpt]およびプログラムの署名SK' s [EKp [Prog]]とをクライアント装置91に送信する。クライアント装置91は受信した領収書と署名を保存するとともに、シーケンス1010で、サーバ93によるプログラムの署名SK' s [EKp [Prog]]をマイクロプロセッサ901に送る。

【0105】マイクロプロセッサ901は、クライアント装置91から受け取ったプログラムの書名SK' s [EKp [Prog]]をサーバ93の公開鍵K' pで検証し、正当であれば、シーケンス1011で暗号化された実行プログラムEKp [Prog]をクライアント装置91に渡す。クライアント装置91はEKp [Prog]を受け取ると、シーケンス1012でサーバ93との間の回線の終了処理を行う。

【0106】以後のマイクロプロセッサ901における暗号化プログラムの実行は、第1実施形態と同様である。

【0107】図11は、図10におけるクライアント装置91の処理フローをしめすフローチャートである。この処理フローは図10のシーケンスで説明したとおりであるが、確認のため簡単に説明する。

【0108】まず、ステップS1101でサーバ93との間に第1の通信路を設定する。ステップS1103で、サーバ93に対してプログラムのダウンロードの要求を行い、上述した認証処理を行う。ステップS1105でマイクロプロセッサ901とサーバ93との間の認証を行い、プログラムをマイクロプロセッサ901に転送する。ステップS1107で、マイクロプロセッサ901から受信終了を受け取る。ステップS1109で、サーバ93との間の課金処理を行い、ステップS1111で、サーバ93から領収書と署名を受信する。ステップS1113で、マイクロプロセッサ901に領収書と署名を渡し、ステップS1115で、マイクロプロセッサ901から暗号化されたプログラムを受け取る。ステップS1117で、サーバ93との間の回線を終了する。

【0109】図12は、図10におけるサーバ93の処理フローを示すフローチャートである。まず、ステップS1201で、クライアント装置91からの要求による第1通信路を設定する。ステップS1203で、クライアント装置91からのプログラムのダウンロード要求を受信し、課金などの目的でクライアントとの認証を行う。ステップS1205で、マイクロプロセッサ901との間に第2の通信路を設定する。ステップS1207で、マイクロプロセッサ901からのチャレンジRcと公開鍵Kpを受信する。ステップS1209で、サーバ側のチャレンジRsとレスポンスS [Rc]を生成し、サーバ93の公開鍵K' pとともにRc、S [Rc]をマイクロプロセッサ901に送信する。ステップS121

1で、マイクロプロセッサ901からのレスポンスS [Rc]と、証明書E [Cert]を受信する。ステップS1213で、マイクロプロセッサ901から受信したS [Rc]とE [Cert]が正当かどうか検証する。検証に失敗した場合は、処理を中止する。検証に成功した場合は、ステップS1215でマイクロプロセッサ901に対応した暗号化プログラムと署名を生成して、それらをマイクロプロセッサ901に送信する。ステップS1217で、マイクロプロセッサ901との間のプログラム転送を終了する。ステップS1219で、クライアント装置91との間で課金処理を行い、領収書、署名を送信する。ステップS1221で、クライアント装置91との間の回線を終了する。

【0110】図13は、図10におけるマイクロプロセッサ901の処理フローを示すフローチャートである。まず、ステップS1301で、サーバ93との間に第2の通信路を設定する。ステップS1303で、チャレンジRcを生成し、マイクロプロセッサ901に固有の公開鍵Kpとともに、サーバ93に送信する。送信が成功したなら、ステップS1305で、サーバ93からレスポンスS [Rc]と、チャレンジRsおよび公開鍵K' pを受信する。ステップS1307で、レスポンスが正当かどうかを判断する。正当でないとした場合は、処理を中止する。正当な場合は、ステップS1309で、サーバ93に対するレスポンスと、サーバに対応した証明書を生成し、送信する。ステップS1311で、サーバ93から暗号化されたプログラムを受信する。受信が終了すると、ステップS1313で、クライアント装置91にダウンロードの終了を通知する。ステップS1315で、クライアント装置91から署名を受信し、ステップS1317で、署名を検証する。検証が失敗すれば処理は中止する。検証に成功すれば、ステップS1319で、暗号化されたプログラムをクライアント装置91に転送する。

【0111】図10のシーケンスにおいて、課金処理が完了した後のダウンロードプログラムの異常終了によって、暗号化されたプログラムE [Prog]が入手できなかった場合は、クライアントは保存した領収書をもとに、再度サーバ93からプログラムをダウンロードする権利を保持する。この場合、当然課金は行われず、このマイクロプロセッサ901をターゲットとした暗号化プログラム以外のプログラムを得ることもない。

【0112】また、図10のプログラム配布シーケンスにおいて、サーバ93が、プログラムの暗号化に、マイクロプロセッサ901の公開鍵を使用せず、共通鍵を使用してもよい。この場合は、サーバ93が共通鍵Kxと暗号化アルゴリズムを選択し、共通鍵Kxでプログラムを暗号化した上で、この共通鍵をプロセッサに固有の公開鍵で暗号化し、サーバ93とマイクロプロセッサ901との間の暗号化を行ってから、プログラムをマイクロ

プロセッサ901に送信する。このとき、第2通信路設定後に、マイクロプロセッサ901に備えられている暗号復号化能力をサーバ側からマイクロプロセッサ901に問い合わせるシーケンスを追加して、マイクロプロセッサ901に処理可能な暗号化アルゴリズムの中からサーバ93が暗号化アルゴリズムを選択してもよい。

【0113】実行コードの復号化にプログラムごとに使い捨てる共通鍵を使うので、鍵の長さを短くし、クライアント装置91のメモリあるいはマイクロプロセッサ901のキャッシュの中に構築されるページテーブルのサイズを小さくできる。

【0114】上述したようなダウンロードの手順は、データベースや顧客情報などの秘密情報の取り扱いにも適用できる。また、プログラムをターゲットのマイクロプロセッサだけで実行できるように暗号化することを除けば、実行プログラム以外の任意の音楽、映像データにも拡張して適用可能である。

【0115】

【発明の効果】本発明のマイクロプロセッサによれば、マイクロプロセッサが組み込まれたPCの所有者に解析されることがなく、暗号化されたプログラムを安全に実行することができる。また、既存のプログラムにソース変更や再コンパイルすることなく、完全にプロセッサ内部で復号化を行うことができる。

【0116】本発明の別の形態のマイクロプロセッサによれば、プログラムの暗号化に共通鍵を用い、共通鍵自体をマイクロプロセッサの公開鍵で暗号化した。これにより暗号化プログラムのセキュリティが補償されるとともに、マイクロプロセッサのハードウェアサイズを大幅に縮小することができる。

【0117】本発明のプログラム配布システムでは、ネットワークを介して、プログラム配布装置からクライアント装置へ、安全かつ確実に暗号化プログラムを配布することができる。また、マイクロプロセッサで実行される耐タンパなダウンロードプログラムの存在を前提とすることにより、第三者を介さず、直接プログラム配布装置とマイクロプロセッサとの間で、プログラムのダウンロードを安全かつ効率良く行うことができる。

【0118】また、コンピュータ読み取り可能な記録媒体に、プログラム本体部分を暗号化して記録し、外部のプログラムへの直接の参照を行うIAT領域は暗号化せずに平文の状態で格納することにより、プログラム実行時のリロケーションが正しく行われることを可能にする。

【図面の簡単な説明】

【図1】本発明の第1実施形態に係るマイクロプロセッサの概略ブロック図である。

【図2】本発明の第1実施形態に係るマイクロプロセッサで実行される暗号化プログラムのファイル形式の一例を示す図である。

【図3】本発明の第1実施形態に係るマイクロプロセッサの物理アドレスと、仮想記憶を管理するページテーブルと、論理アドレスとの対応関係を示す図である。

【図4】図3に示すページテーブルの中の、ページエントリの一例を示す図である。

【図5】本発明の第2実施形態に係るマイクロプロセッサの概略ブロック図である。

【図6】本発明の第2実施形態に係るマイクロプロセッサで実行される暗号化プログラムのファイル形式の一例を示す図である。

【図7】本発明の第2実施形態に係るマイクロプロセッサの物理アドレスと、仮想記憶を管理するページテーブルと、キーテーブルとの対応関係を示す図である。

【図8】図7に示すページテーブルとキーテーブルとの関係を詳細に示す図である。

【図9】本発明の第3実施形態に係る、プログラム配布システムの図である。

【図10】本発明の第3実施形態に係るプログラム配布のシーケンス図である。

【図11】図9に示すクライアント装置のプログラム受け取り手順を示すフローチャートである。

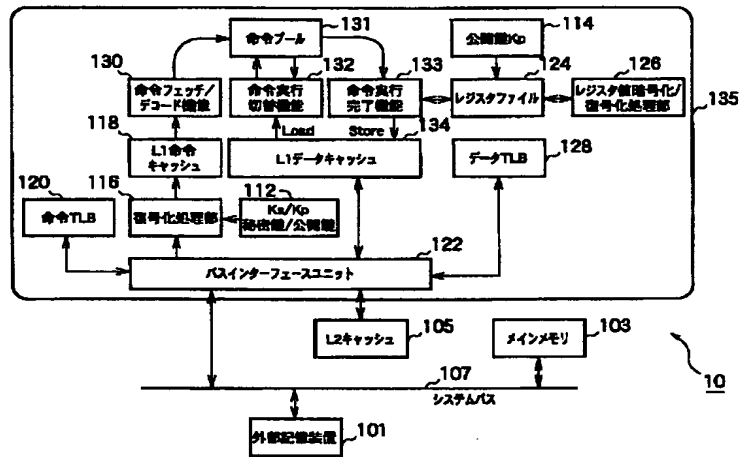
【図12】図9に示すプログラム配布装置のプログラム配布手順を示すフローチャートである。

【図13】図9に示すクライアント装置内部のマイクロプロセッサの動作を示すフローチャートである。

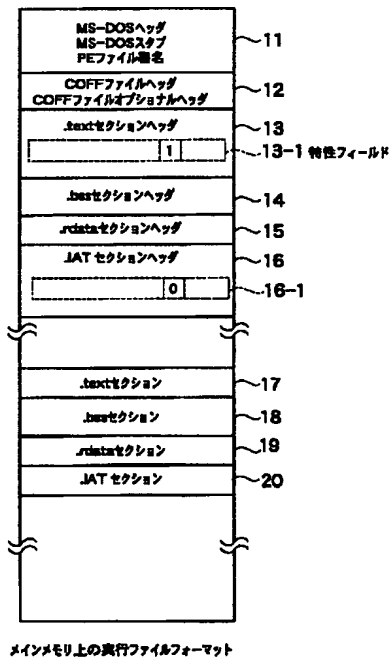
【符号の説明】

- 1 0、2 0、9 0 1 マイクロプロセッサ
- 3 7、7 7 ページテーブル
- 7 9 キーテーブル
- 9 1 クライアント装置
- 9 3 プログラム配布装置
- 9 5 ネットワーク
- 1 0 3、2 0 3 メインメモリ
- 1 0 5、2 0 5 二次キャッシュ
- 1 1 2、2 1 2 秘密鍵
- 1 1 4、2 1 4 公開鍵
- 1 1 6、2 1 6 復号化処理部
- 1 1 8、2 1 8 一次命令キャッシュ（第2の記憶手段）
- 1 2 0、2 2 0 命令TLB（第1の記憶手段）
- 1 2 2、2 2 2 バスインターフェイスユニット
- 1 2 4、2 2 4 レジスタ
- 1 2 6、2 2 6 レジスタ値暗号化／復号化処理部
- 2 3 6 復号化鍵TLB
- 9 0 5、9 3 2 第1通信設定部
- 9 0 6、9 3 4 第2通信設定部
- 9 1 0、9 3 3 ユーザ認証部
- 9 0 8、9 1 4 プログラム受信部
- 9 3 5 プログラム認証部
- 9 3 6 プログラム暗号化処理部

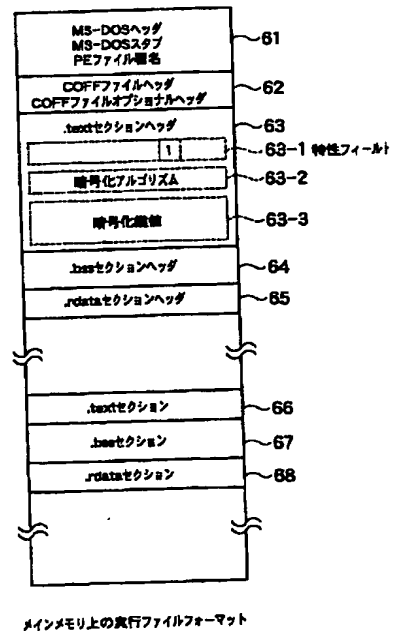
【図1】



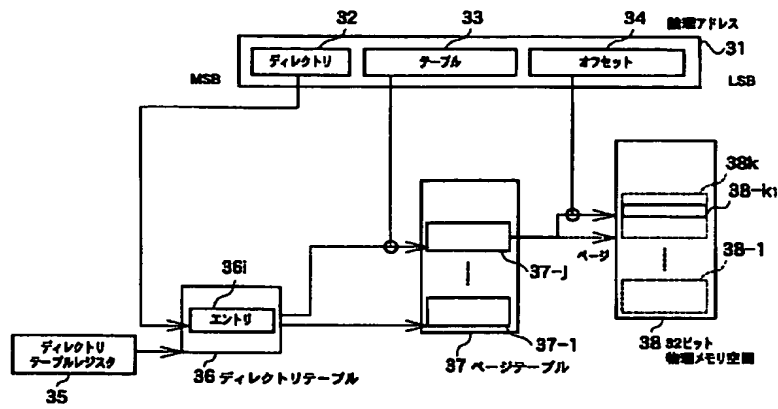
【図2】



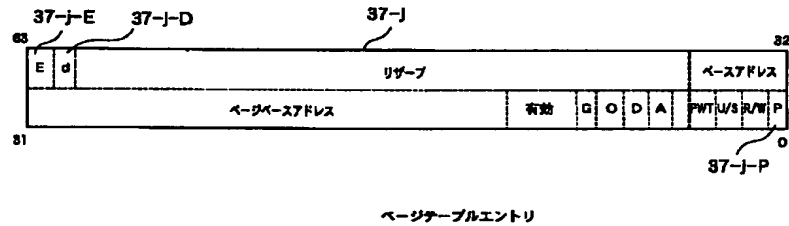
【図6】



【図3】

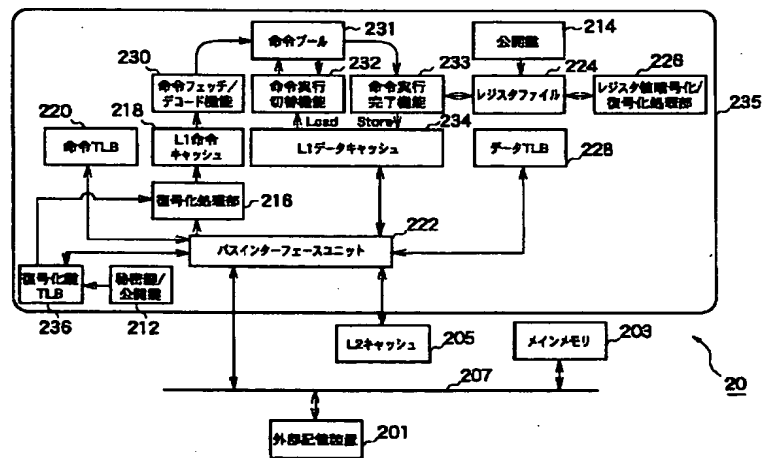


【図4】

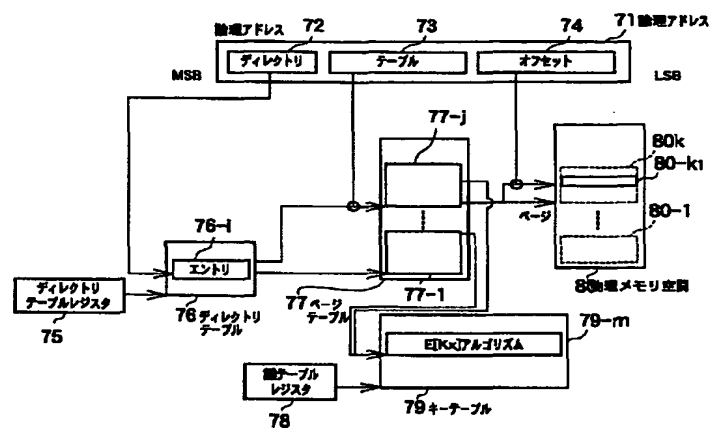


ページテーブルエントリ

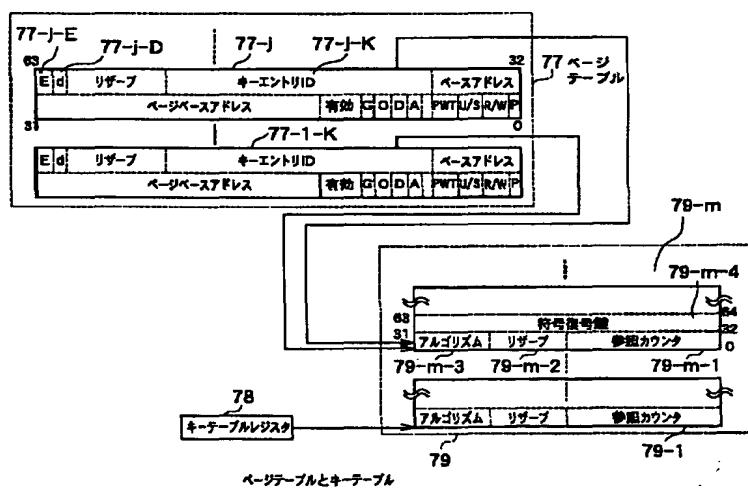
【図5】



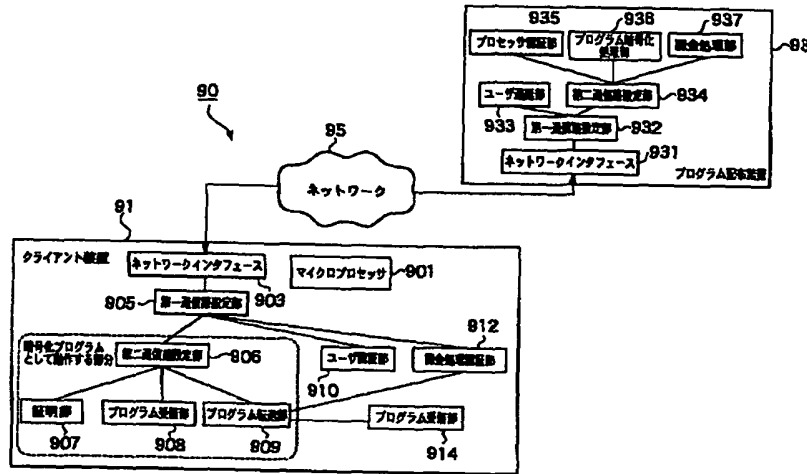
【図7】



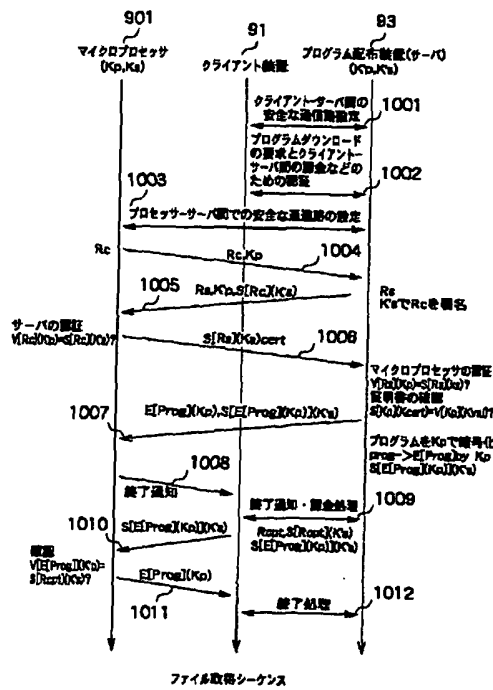
【图 8】



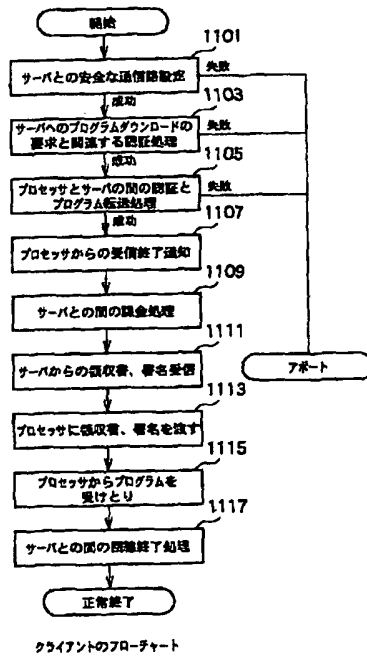
【図9】



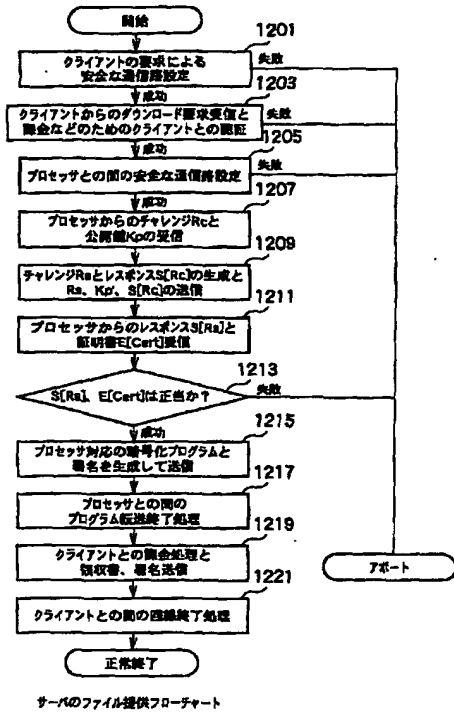
【図10】



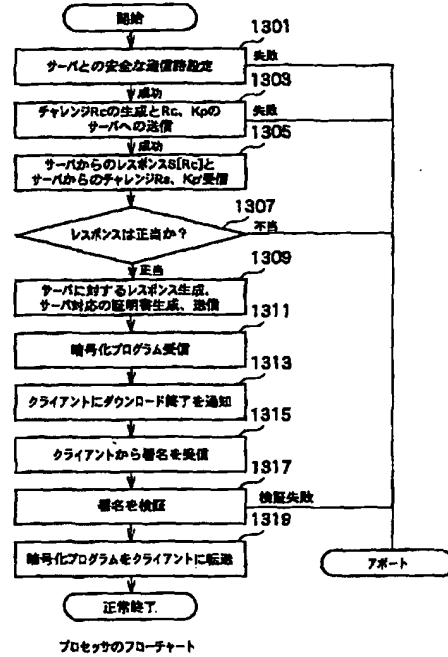
【図11】



【図12】



【図13】



フロントページの続き

(72)発明者 寺本 圭一
 神奈川県川崎市幸区小向東芝町1 株式会社
 社東芝研究開発センター内

Fターム(参考) 5J104 AA12 JA21 KA06 MA01 NA02
 NA03 NA12 NA22 NA24 NA27
 PA07
 9A001 BB03 BB04 EE03 LL03